

# **How to Dedupe**

# **Rows in SQL**

# **WITHOUT Using**

# **DISTINCT**



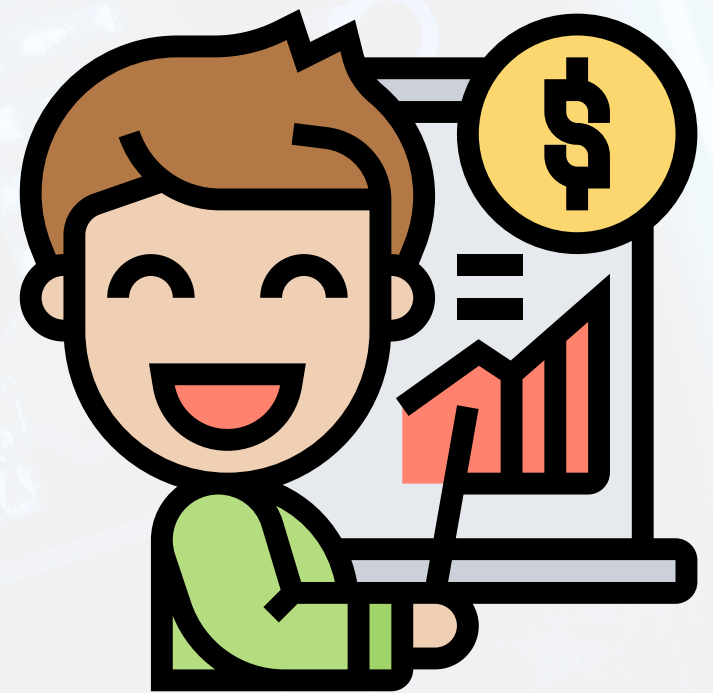


**Duplicate rows can  
be a nightmare  
for data  
analysts.**



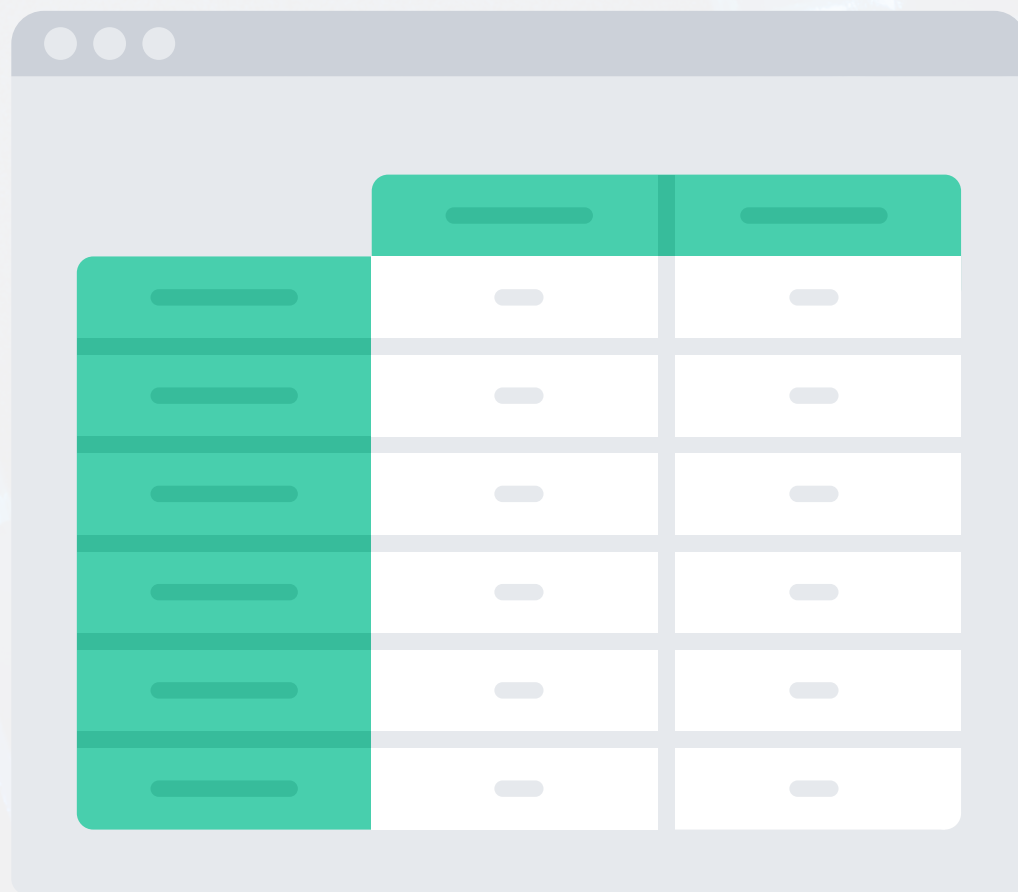


In this guide, I  
will show you  
**an elegant**  
way to fix  
them.



# The DISTINCT

keyword is often  
used to remove  
duplicates.

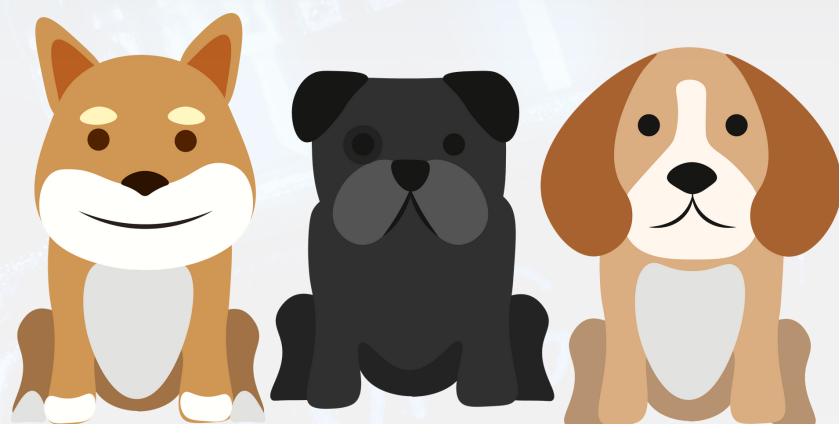




Suppose you have a table  
named **animals** with the  
following data...

id	species	color
1	dog	brown
2	cat	black
3	dog	white
4	cat	white
5	dog	brown

← Dupes






# To dedupe with DISTINCT, you would do this...



```
SELECT DISTINCT species, color  
FROM animals
```

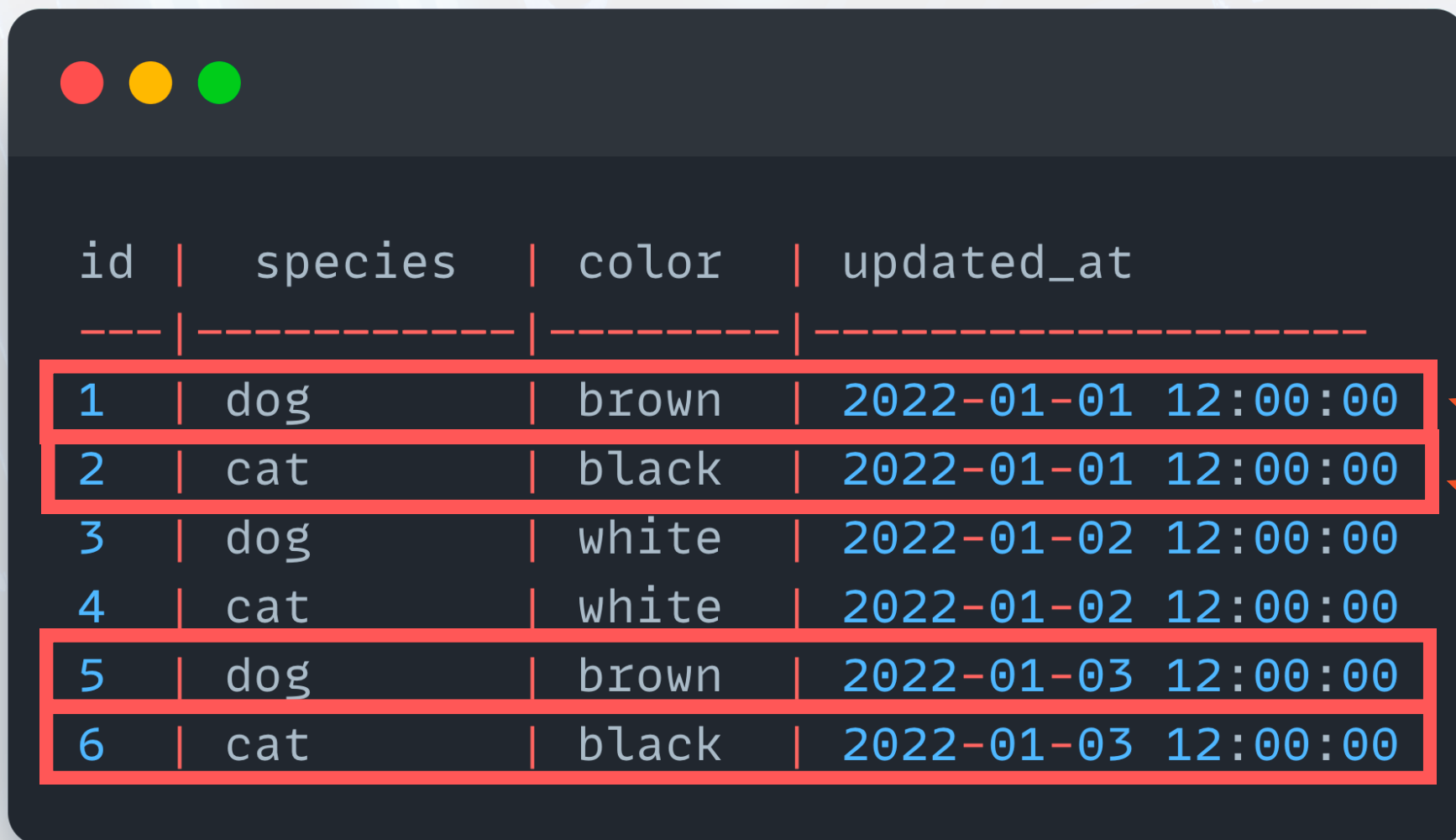
## Output:



species	color
dog	brown
dog	white
cat	black
cat	white



But what if your data looks like *this*...

A terminal window with a dark background and three colored window control buttons (red, yellow, green) in the top-left corner. It displays a table with four columns: 'id', 'species', 'color', and 'updated\_at'. The table has six rows of data. The first, second, fifth, and sixth rows are highlighted with red borders. To the right of the table, three orange arrows point to these highlighted rows, with the word 'Dupes' next to them.

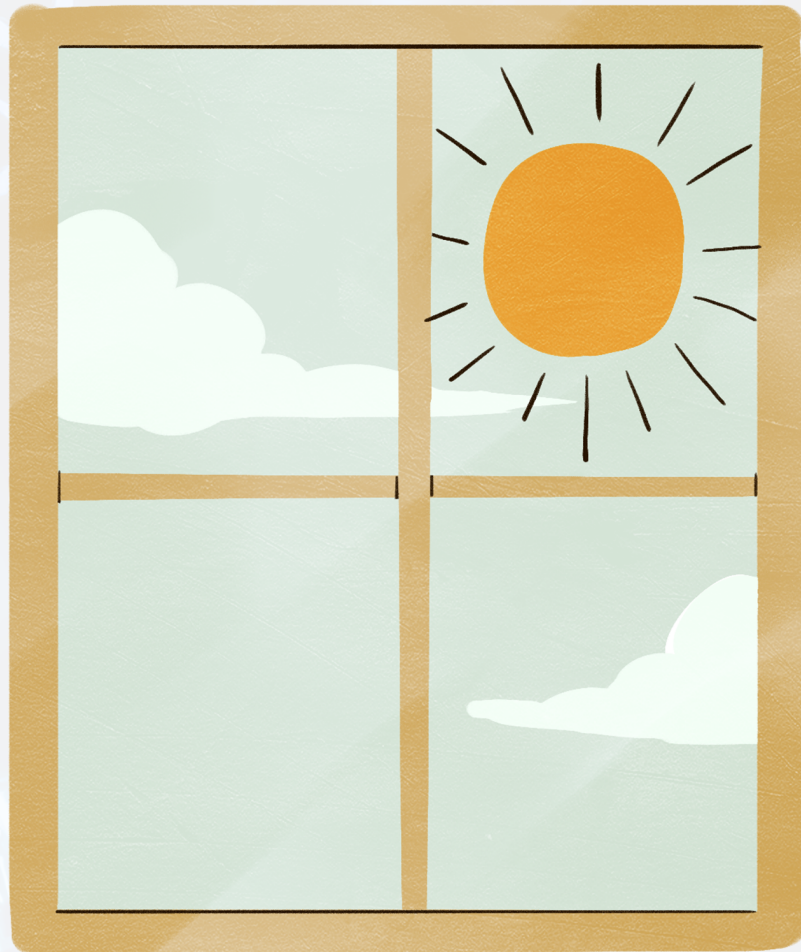
id	species	color	updated_at
1	dog	brown	2022-01-01 12:00:00
2	cat	black	2022-01-01 12:00:00
3	dog	white	2022-01-02 12:00:00
4	cat	white	2022-01-02 12:00:00
5	dog	brown	2022-01-03 12:00:00
6	cat	black	2022-01-03 12:00:00

Dupes

And you want the  
**most recent** updated  
value for each?



# Enter Window Functions!



The window function we  
will use is called:

**ROW\_NUMBER()**





Using **ROW\_NUMBER()**  
allows us to be more  
**precise** with our  
deduplication.



# To use `ROW_NUMBER()` start with this...

```
SELECT
  *,
  ROW_NUMBER() OVER (PARTITION BY species, color
                     ORDER BY updated_at DESC) AS row_num
FROM animals
```

## Output:

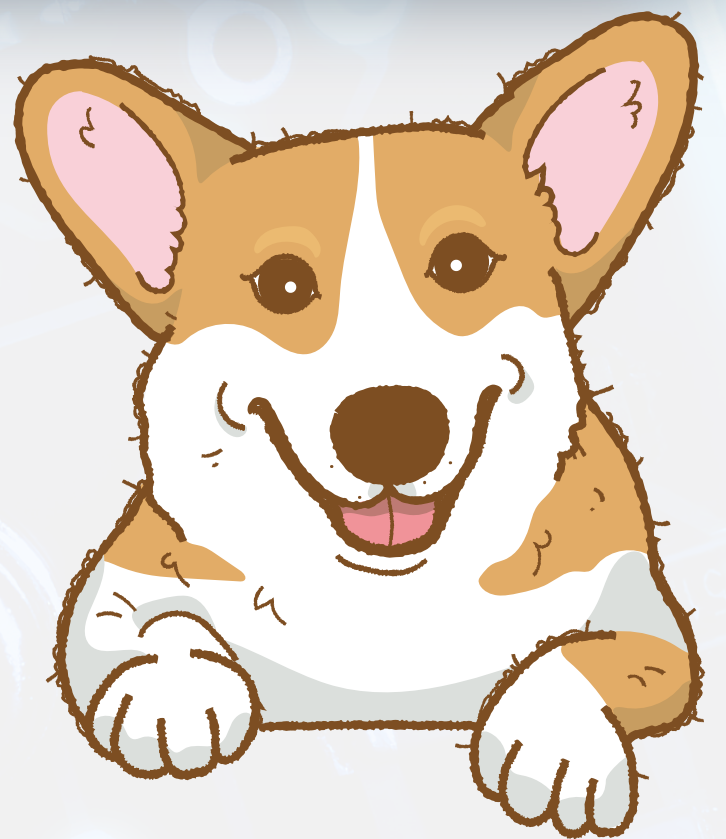
id	species	color	updated_at	row_num
5	dog	brown	2022-01-03 12:00:00	1
1	dog	brown	2022-01-01 12:00:00	2
3	dog	white	2022-01-02 12:00:00	1
4	cat	white	2022-01-02 12:00:00	1
6	cat	black	2022-01-03 12:00:00	1
2	cat	black	2022-01-01 12:00:00	2



# Then throw it in a CTE

where row\_num = 1

```
WITH cte AS (  
    SELECT  
        *,  
        ROW_NUMBER() OVER (PARTITION BY species, color  
                             ORDER BY updated_at DESC) AS row_num  
    FROM animals  
)  
SELECT *  
FROM cte  
WHERE row_num = 1
```



# BOOM!



## Deduplicated Output:

id	species	color	updated_at	row_num
3	dog	white	2022-01-02 12:00:00	1
4	cat	white	2022-01-02 12:00:00	1
5	dog	brown	2022-01-03 12:00:00	1
6	cat	black	2022-01-03 12:00:00	1



Practice using

**ROW\_NUMBER()**

and you will have it

mastered in

no time.



Duplicates can lead  
to **inaccuracies** in  
your data.








Now you know a  
**cool** way to fix  
them!





# Like this content?

Every week I send a free email with:

- **SQL tips** 
- **Example projects** 
- **Data memes** 

Link to sign up in bio.



@Kyle Malone, CFA